1 Name of Use Case

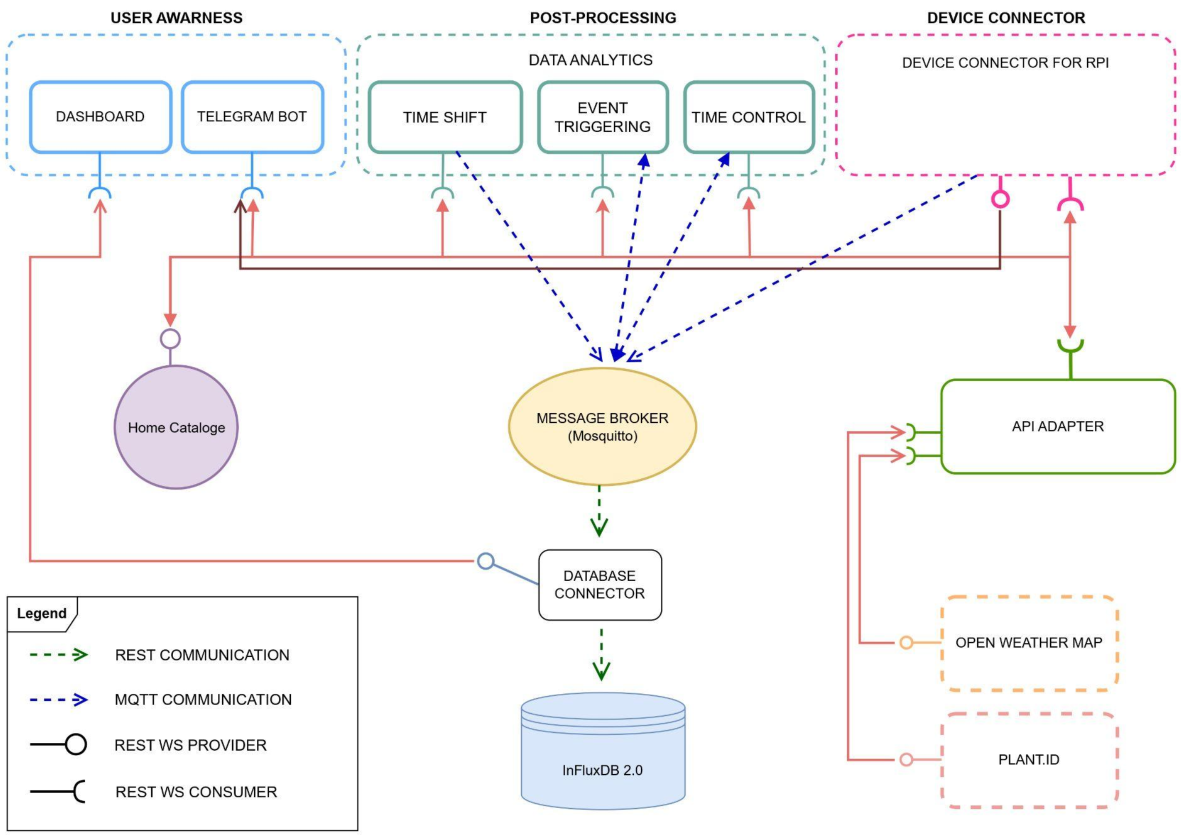
Project Name	Smart Garden IoT Project
Version No.	v0.2
Submission Date	06.01.2024
Team Members	Wenxi Lai (S326332) Davood Shaterzadeh (S325715) Nasrin Hayati nejad (S328272) Kamand Aghaeiabri (S327554)

2 Scope and Objectives of Function

Scope and Objectives of Use Case	
Scope	The Smart Garden IoT project aims to create an intelligent and automated gardening system that leverages IoT technologies for monitoring and managing the garden environment. The scope includes the integration of sensors, actuators, and a central control system using Raspberry Pi.
Objective(s)	• Monitor soil moisture, temperature, humidity, and light conditions in real-time. • Automate watering based on soil moisture levels. • Implement a time-control feature for scheduled watering. • Integrate Weather Forecasts to anticipate upcoming weather conditions. • Provide remote monitoring and control capabilities via a user-friendly interface. • Implement alerts and notifications for critical environmental conditions. • Enable expansion for additional sensors or actuators based on user needs.
Domain(s)	• Environmental Monitoring • Smart Agriculture
Stakeholder(s)	• Homeowners • Gardening Enthusiasts
Short description	The Smart Garden IoT, powered by Raspberry Pi, transforms gardening with IoT innovation. Monitoring soil conditions, automated watering, and a user-friendly interface enhance plant care. The time-control feature enables scheduled watering, adding flexibility. Integrated Weather Forecast anticipates conditions for informed decisions. Customized for homeowners and gardening enthusiasts, this project seamlessly combines tech and gardening, offering efficiency and a holistic experience. Simple, effective, and scalable, it presents a novel, connected approach to nurturing plants. • Redefines gardening with perfect IoT integration • Sensors and Raspberry Pi enable real-time monitoring • Monitors soil conditions, temperature, humidity, and light levels

	● Automated watering optimizes plant health ● User-friendly remote interface for personalized control ● Alerts for critical conditions enhance plant care ● Customized for homeowners and gardening enthusiasts ● Rhythmic blends technology with gardening ● Promotes a smarter, connected, and engaging plant care experience

3 Diagram of Use Case



---

4 Complete Description

The system consists of sensors to measure soil moisture, temperature, humidity, and light conditions. These sensors are connected to a Raspberry Pi, the central hub. The Raspberry Pi processes the data, makes decisions based on predefined rules, and controls actuators such as water pumps for automated watering. The time-control feature allows users to schedule watering sessions at specific times, providing flexibility and efficient water management.

The system's participants have been pinpointed and are elucidated as follows:

RASPBERRY Pi

Our team's IoT project relies on the Raspberry Pi, a flexible and affordable edge computing device. It seamlessly integrates various sensors, processes data locally, and makes real-time decisions. As a communication hub, it facilitates efficient data exchange among connected devices and the central system. With remote access, users can monitor and manage the IoT environment from anywhere. The Raspberry Pi extends project capabilities by integrating external services, like weather forecasts or plant databases, and supporting edge analytics for immediate insights. Robust security measures, including encryption and access controls, safeguard local data and communications. In summary, the Raspberry Pi is instrumental in enhancing our IoT project's efficiency, responsiveness, and overall functionality.

MESSAGE BROKER

In our smart gardening IoT project, Mosquitto, a robust message broker, plays a crucial role in enabling seamless communication among devices, sensors, and backend systems. This strategic choice empowers our IoT ecosystem, fostering a responsive, scalable, and reliable communication infrastructure tailored to the unique requirements of our smart gardening project.

HOME CATALOGUE

The home catalogue serves as an exhaustive hub containing essential details about system endpoints and initial configurations, which are crucial for launching the entire IoT ecosystem. Housing REST and MQTT topics for devices and resources, it not only stores but orchestrates valuable starting configurations. This centralized home catalogue ensures seamless system initiation and provides a comprehensive reference point for actors within the IoT network, streamlining the integration of services and devices.

POST-PROCESSING MICROSERVICES

The smart garden system employs a set of post-processing microservices to analyze sensor data, identify patterns, and trigger appropriate actions. These microservices utilize sophisticated algorithms to go beyond simple threshold-based triggering and provide more intelligent decision-making.

Event Triggering:

The event-triggering microservice monitors sensor data for predefined events or conditions. For instance, it checks if soil moisture levels drop below a certain threshold, indicating the need for irrigation. Similarly, it detects extreme weather conditions, such as heat waves, and triggers actions to conserve water or adjust lighting settings. User interactions, such as setting preferences for light exposure or watering frequency, can also trigger events.

METHODOLOGY

Inputs	Sensor data (moisture levels, temperature, light intensity), user-defined preferences
Algorithm architecture	Leverages statistical analysis and machine learning techniques
Output	Control signals for watering mechanisms, adjustments in light exposure, and user notifications
Details	Statistical analysis: Identifies trends, patterns, and anomalies in sensor data
Machine learning algorithm	Predict future behaviour, optimize watering schedules, and personalize plant care

Time Shift:

The time shift microservice optimizes the timing of actions within the IoT system to minimize resource consumption and maximize efficiency. It schedules data collection during off-peak hours when energy costs are lower and adjusts watering schedules to avoid peak sunlight hours.

METHODOLOGY

Input	Current time, energy cost data, and historical patterns
Algorithm architecture	Utilizes time series analysis and scheduling techniques
Output	Scheduled tasks, adjusted timing for specific actions
Details	Time series analysis: Forecasts future energy costs and identifies optimal data collection times
Scheduling techniques	Optimizes watering schedules based on weather conditions, energy costs, and sensor data

Time Control

The time control microservice governs system behavior based on specific time-related parameters. It schedules routine actions, triggers events based on the current time or predefined intervals, and simulates natural light cycles for plant growth.

METHODOLOGY

Input	Current time, predefined schedules, and environmental conditions
Algorithm architecture	Employs decision trees, rule-based systems, and time-series analysis
Output	Scheduled actions, triggered events, and adjustments in system behaviour
Details	Decision trees: identify optimal watering and lighting schedules based on environmental conditions and historical data
Rule-based systems	Enforce predefined schedules and respond to specific events
Time-series analysis	Predicts future plant growth patterns and optimizes lighting cycles

API ADAPTER

The API adaptor plays a key role in our system, connecting it to external services and providing alerts if APIs become unavailable. It ensures the system stays scalable. The two primary APIs initially connected to the system are:

- **Plant.id**

  Upload a picture of a plant to enhance the garden database. Information includes Wikipedia links, name, origin, and care advice.

- **OpenWeatherMap**

  This integration adds intelligence by connecting to a weather forecasting service. It helps anticipate weather conditions, influencing gardening decisions like adjusting watering schedules or protecting plants during frost warnings.

USER AWARENESS

The user interface, accessible both remotely via the internet and locally, serves as the ultimate endpoint for users. It enables real-time monitoring, preference settings, scheduling watering times, and access to weather forecasts. The system prioritizes security measures to safeguard against unauthorized access, ensuring a seamless and secure gardening experience. Users can stay informed about weather forecasts, historical data, analyses, advice, and current conditions such as humidity, temperature, and soil moisture—all contributing to a comprehensive user awareness of the garden's well-being.

- **Dashboard**

  Is a user-friendly control center, offering comprehensive details and access to the database. It displays sensor histories and graphs, providing a clear understanding of the garden's conditions over time.

- **Telegram bot**

  offers real-time updates on temperature, humidity, and soil moisture. Using system data, it summarizes the garden's situation and can remotely activate sprinkler valves or modify watering schedules via the Internet.

DATABASE

InfluxDB emerges as an optimal choice for our IoT project, offering a robust and scalable solution for managing time-series data. As part of our deployment strategy, we intend to leverage Docker to containerize InfluxDB, making it the central database for our smart garden system. Containerization enhances portability and facilitates efficient storage and retrieval of time-series data within the IoT ecosystem. This strategic decision ensures a seamless and resource-efficient database solution, aligning perfectly with the demands of our project.

DATABASE CONNECTOR

The Database Connector microservice acts as a crucial intermediary in our smart gardening IoT project, facilitating seamless communication between the application and InfluxDB2. Key features include dynamic configuration for future database transitions, periodic health checks to ensure accessibility, a failover mechanism for database outages, and robust logging and monitoring for performance insights. This microservice ensures adaptability, resilience, and secure database interactions, contributing to the overall reliability of our IoT ecosystem.

5 Desired Hardware Components (only among those we can provide)

Device Name	Quantity	Needed for…
Raspberry Pi (Model 2)	1	Central Processing and Control
Soil Moisture Sensor	1	Monitoring Soil Moisture Levels
Temperature/Humidity Sensor	1	Monitoring Ambient Conditions
Light Sensor	1	Monitoring Light Exposure
Water Pump	1	Automated Watering (Model)
sprinkler valve	1	Automated Watering (Model)
Relay	1	Controlling Power to Actuators
Wi-Fi Module	1	Internet Connectivity